REMARKS

Claim Amendments

Applicants appreciate the Examiner's consideration of the present application. The present application contains six independent claims 1, 9, 35, 36, 43 and 44.

Applicants have amended independent claims 1, 35, 43 and 44 by clarifying that the metadata model has a lower layer containing one or more model objects having a lower degree of abstraction and a higher layer containing one or more model objects having a higher degree of abstraction. Applicants have further amended these claims by reciting that information is obtained from a model object in the lower layer, and that a model object is created in the higher layer corresponding to the model object in the lower layer. These amendments are made to clarify the claim language and improve the readability of the claims.

Applicants have also amended dependent claims 2-8 in view of the amendments made to parent claim 1.

In claim 22, Applicants have amended "entities that exist as an implementation artifact of a many to many relationship, and many to many business joins associated with the entities" to --entities that exist as an implementation artifact of a many to many relationship, and business joins associated with the entities--, and "the associated many to many business joins" to --the associated business joins--. These amendments are made to clarify that the business joins are associated with the entities "that exist as an implementation artifact of a many to many relationship", rather than they are always "many to many".

Claims 40 and 41 have been amended that the new package layer is constructed based on the business model objects. This amendment is based on the

description of the package model constructing transformations on pages 56-58 of the specification as originally filed.

Also, Applicants have amended claims 9, 11, 13, 26 to correct typographical errors. The amendment to claim 13 brings the package model transformations in line with the data access model transformations and business model transformations recited in claims 9.

It is respectfully submitted that no new matter has been introduced into the application by the above amendments. Pursuant to 37 CFR 1.121, a marked copy of the amended claims showing the changes made therein accompanied this amendment.

Rejection under 35 USC 103(a) to claims 1-8, 35 and 43-44

The Examiner has rejected claims 1-8, 35 and 43-44 under 35 U.S.C. 103(a) stating that these claims are unpatentable over Brunner et al (US Patent No. 5,550,971) in view of Pouschine et al (US Patent No. 5,918,232). Applicants respectfully request reconsideration of the rejection based on the reasons set out below.

The present invention as recited in independent claims 1, 35, 43 and 44 transforms a metadata model having lower and higher layers by creating a model object in a higher layer based on a model object in a lower layer.

Brunner et al discloses a method and system for generating a user interface. Brunner's system uses an intermediate model "as support for the user interface design and generation..... The data model is a semantic model that is dynamically analyzed in order to generate the user interface." (column 3, line 62-column 4, line 5). Column 5, lines 20-31 describes the semantic data model and its three layers referring to Figure 2. The "most fundamental level" (column 5,

line 36), the meta-meta-model layer, and the meta-model layer are the layers of programs used to implement the model layer. The program code of the meta-meta-model layer and the meta-model layer is shown from column 13, line 61 to column 14, line 45. To create a database model corresponding to the model shown in Figure 2, as shown in Figure 3, a new instance of the DataModel class is created at step 100, and the standard data types of the meta-meta-model layer and the meta-model layer are added at step 102 (column 13, lines 46-55).

By contrast, the term "layers" is used in the present application to refer to the internal structure of the metadata. The layers of the metadata model referred to in the present application reside within Brunner's meta-model layer. Appendix A, appended to this Response, illustrates these correspondences. Brunner's model has three layers: model layer, meta-model layer and meta-meta-model layer, which are provided on top of another. If the metadata model of the present invention as exemplified in claim 9 is to be illustrated in the same manner as Figure 2 of Brunner, the data access layer, business layer and package layer would be arranged side by side as shown in Appendix A because those layers reside within each of Brunner's model layer and meta-model layer. Thus, the term "layers" as used in Brunner refers to a different concept from the term "layer" used in the present application. In order to simplify the discussion, we will refer to Brunner's layers as "stages" in the following paragraphs as Brunner's layers represent programming and implementation stages.

In the example shown in Appendix A, Table in the data access layer, Entity in the business layer and Subject in the package layer all correspond to Entity and Scalar in the meta-model stage of Brunner. Table in the data access layer is instantiated as COURSE, STUDENT_IN_COURSE and STUDENT. These instances correspond to instances shown in the model stage of Brunner. Similarly, instances of Entity in the business layer include Courses and Students, which correspond to instances shown in the model stage of Brunner, and so on.

Each stage in Brunner's model has a single structure. Brunner does not disclose any layers within a stage.

The present invention is directed to the transformation of objects between layers in a metadata model (e.g., claims 1 and 9) or within a layer of the metadata model (e.g., claim 9)). As recited in claim 1, a model object in a layer with a lower degree of abstraction is transformed and a new model object is created in a higher layer with a higher degree of abstraction. Thus, an object in a source database is represented as multiple model objects in the metadata model, e.g., "course" in a source database is represented as COURSE in the data access layer, as Courses in the business layer and as Courses in the package layer as shown in Appendix A. By contrast, Brunner et al does not disclose any transformation of objects within a stage or creation of new objects within the same stage for representing the same object or table in its underlying database. As shown in Figure 3, step 104, only a single model object is created per object in the database, e.g., COURSE is the only model object that represents "course" in the database as shown in Appendix A. There is no other object transformed based on COURSE in the data stage. Similarly, each of CLIENT, PATENT and LITIGATION shown in Figure 2 of Brunner is only a single model object that represents its respect table in the underlying database.

Therefore, Brunner does not disclose or suggest any metadata model transformer as claimed in the claims.

The Examiner has indicated that Brunner does not teach business intelligence, and used Pouschine et al to cover this aspect.

Pouschine et al discloses the use of "a rule-based methodology" as the Examiner pointed out. These rules define a methodology for calculating values from data in the database, rather than a methodology for transforming model objects in a metadata model, as further discussed below.

Therefore, even if one skilled in the art combines Brunner et al and Pouschine et al, he would still fail to provide a metadata model transformer or method for transforming a metadata model. Thus, the present invention as recited in independent claims 1, 35, 43 and 44 and dependent claims 2-8 have been patentably distinguished over Brunner et al and Pouschine et al.

Rejection under 35 USC 103(a) to claims 9-21, 24-33 and 36-42

The Examiner has also rejected claims 9-21, 24-33 and 36-42 under 35 U.S.C. 103(a) stating that these claims are unpatentable over Pouschine et al (US Patent No. 5,918,232) in view of Mellen-Garnett et al (US Patent No. 6,094,688). Applicants respectfully request reconsideration of the rejection based on the reasons set out below.

Pouschine et al discloses a modeling system that does not require pre-calculation of all data cells and allows data to remain in database warehouses until needed (column 3, lines 61-64). It creates multidimensional data models or hyperstructures. Pouschine obtains measurements of physical objects and activities which are related to the entity to be modeled in the computer hyperstructure, and "transform" these measurements into computer data which corresponds to the physical objects and activities external to the computer system (column 4, lines 23-31). That is, the transformation described by Pouschine is manipulation of measurements of physical objects and activities, and not abstraction or refinement of model objects.

To create hyperstructures, Pouschine uses a Distributed On-Line Analytical Processor (DOLAP). DOLAP creates elements dynamically by using the dynamic elements (column 10, lines 65-67). Column 7, lines 13-14, a definition of "dynamic element specification" is provided as "a description of how dynamic elements are created from the database on demand". "DOLAP extracts the

elements from the tables" (column 11, line 1). From these descriptions, it is clear that dynamic elements are created on demand during querying data from the database, rather than during the transformation or refining of model objects to construct a metadata model. Thus, Pouschine discloses levels of dynamic hierarchy on column 11, lines 19-27, but those levels do not correspond to the layers of the metadata model recited in the claims of the present application. Furthermore, Pouschine describes an example of creation of a customer hierarchy using a database table in the same section on column 11, but it does not disclose how a model is created from the database tables. Thus, Pouschine et al does not disclose any layers of metadata model or how to create a metadata model. By contrast, according to the present invention, as recited in independent claims 9 and 36, a metadata model has a data access layer, business layer and a package layer, and model objects in these layers are transformed to create model objects in a higher layer based on model objects in a lower layer.

Pouschine defines rules (column 11, lines 49-61; column 12, lines 1-7). Pouschine discloses "To obtain the values for cells in the model, DOLAP uses two types of rules: 1) formula rules which tell DOLAP to compute the cell value mathematically from other cells in the model ... and 2) data map rules which tell DOLAP to retrieve the cell value from the database" (column 12, lines 1-6). As it is clear from this description, these rules are used to obtain right data values during querying, i.e., to calculate values and populate the values in a model based on data in a database. These rules are not business model objects contained in a business layer of a metadata model or transformation of model objects as recited in claims 9 and 36 of the present application.

Also, in column 14, lines 20-23, Pouschine discloses "a named file-like package". Pouschine's package means a "container" of the model 50, i.e., the model 50 is entirely packaged in a container. Pouschine's package is not a part of the model 50, but rather the model 50 itself. This is clearly described on column 8, lines 1-6

such that "model" itself is defined as a "named, file-like package" consisting of those listed items. By contrast, the package model objects of the present invention are a part of the package layer forming the metadata model.

The Examiner has stated that Pouschine does not teach the refining of the business rules, and used Mellen-Garnett et al to cover this aspect.

Mellen-Garnett et al discloses a modular application collaborator for providing inter-operability between applications operating in an information system. As shown in Figure 1, Mellen-Garnett uses interchange server 20 having an object-oriented model. Applications 70 can communicate with connectors 30 which provide a schema for interacting with other applications in the object-oriented model of interchange server 20 (column 3, lines 46-49). Mellen-Garnett discloses use of "meta-data" in the interchange server's objects, which describes the information that characterizes the data. The meta-data is used so that development tools 60 are driven from the meta-data to allow customers to develop their own application collaboration modules and connectors (column 4, lines 23-28). Mellen-Garrnett does not disclose any transformation of model objects for creation of a metadata model.

In Mellen-Garnnett, the "application collaboration defines the inter-operablility between two or more applications" (column 1, lines 58-60). Mellen-Garnnett discloses data transformation service 246. This is provided to perform both syntactic and semantic transformation of data, such as integer to character conversions to transforming the semantic content and meaning of a term. The transformation is carried out "during a collaboration" of applications (column 7, lines 23-45) so that different applications can communicate. Mellen-Garnnett's transformation is not a transformation of model objects in a metadata model.

The Examiner has pointed out that Mellen-Garnett teaches a screen. Mellen-Garnett's screen is described as an example of rules 1330 provided by the

business encapsulation module 1306 of connector 30 shown in Figure 13 when a filter is stored as a rule for processing by rule engine 1332 (column 19, lines 6, 58-67, column 20, lines 1-7). Mellen-Garnett describes that the screen filters records that are not required for a collaboration and only provides those appropriate records as objects to the application collaboration module 40 (Fig. 1). That is, the screen applies to records containing data, rather than metadata. Also, the filtering is performed during the retrieval of data from the underlying application into the connector 30, rather than on objects constructed from the records within the connector 30. By contrast, according to the present invention as claimed in claim 9, the metadata model transformer transforms and refines model objects within the metadata model, i.e., the transformations apply to objects representing metadata which been imported into or created within the metadata model. Accordingly, Mellen-Garnett's screen is different from business model transformation or its method recited in claims 9 and 36 of the present application.

Moreover, Mellen-Garnett does not teach or suggest any metadata model having a data access layer, business layer and a package layer as recited in claims 9 and 36.

Therefore, even if one skilled in the art combines Pouschine et al and Mellen-Garnett et al, he would still fail to provide a metadata model transformer or method for transforming a metadata model as recited in claims 9 and 36 and their dependent claims. Thus, the present invention as recited in these claims have been patentably distinguished over Pouschine et al and Mellen-Garnett et al.

Rejection under 35 USC 103(a) to claims 22 and 23

The Examiner has further rejected claims 22 and 23 under 35 U.S.C. 103(a) stating that these claims are unpatentable over Pouschine et al (US Patent No.

5,918,232) in view of Mellen-Garnett et al (US Patent No. 6,094,688) and Henninger et al (US Patent No. 5,499,371).

Claims 22 and 23 depend on claim 21 which depends on claim 9. As discussed above, claim 9 has patentably distinguished over Pouschine and Mellen-Garnett.

Henninger et al discloses an apparatus for using an object model of an object-oriented application to automatically map information between an object-oriented application and a structured database, e.g., a relational database. As described on column 8, lines 48-53, for each one-to-one and one-to-many relationship in the object model, a foreign key column or foreign key columns are added to the database table schema in the appropriate table of the database schema, and for each many-to-many relationship in the object model, a separate join table is added to the database schema. In these cases, Henninger's method constructs a database schema and a transform, using the object model as input.

The object model of Henninger is not transformed. As shown in Figures 1 and 3, Henninger's method software 15 accepts object model 20 and accepts or creates database schema 30 and transform 50, and using these three elements as input, the method automatically generates source code (column 5, lines 63-65; column 7, lines 29-31). As seen in Step D of Figure 3, Henninger's method constructs a database schema 30 and a transform 50 derived from the object model 20 (column 7, lines 53-55). Thus, Henninger simply uses the input object model 20, and does not modify or transform the object model 20 as part of the process. This is contrary to the present invention that transforms the metadata model. Therefore, the present invention as claimed is totally different from Henninger.

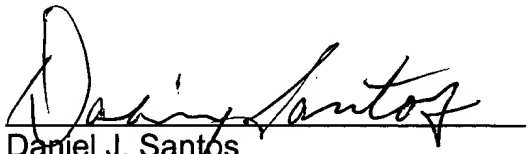
Therefore, claims 22 and 23 are also patentably distinguished over Pouschine, Mellen-Garnett and Henninger.

CONCLUSION

In conclusion, Applicants respectfully submit that the present invention as claimed in the claims is patentably distinguished over any combination of the cited references.

In view of the above amendments and arguments, Applicant respectfully requests reconsideration and early favourable prosecution of the application. Should there be any further questions or concerns, the Examiner is urged to telephone the undersigned to expedite prosecution.

Respectfully submitted,
GARDNER GROFF, P.C.



Daniel J. Santos
Reg. No. 40,158

GARDNER GROFF, P.C.
Paper Mill Village, Building 23
600 Village Trace, Suite 300
Marietta, Georgia 30067
Phone: 770.984.2300
Fax: 770.984.0098

Version Showing Amendments to the Claims as Filed:

1. (Once Amended) A metadata model transformer for transforming a metadata model, the metadata model having a lower layer containing one or more ~~lower abstraction~~ model objects having a lower degree of abstraction ~~level~~ and a higher layer containing one or more ~~higher abstraction~~ model objects having a higher degree of abstraction ~~level~~, the transformer comprising:

a lower-to-higher transformation having:

means for obtaining information from ~~of~~ a ~~lower abstraction~~ model object in ~~from~~ the lower layer;

means for abstracting the information by adding business intelligence; and

means for creating a model object in the higher layer ~~a higher abstraction model object~~ corresponding to the ~~lower abstraction~~ model object in the lower layer.

2. (Once Amended) A metadata model transformer as claimed in claim -1 further comprising:

a lower layer transformation having:

means for obtaining information from ~~of~~ a ~~lower abstraction~~ model object ~~from~~ in the lower layer;

means for modifying the obtained information; and

means for transforming the ~~lower abstraction~~ model object in the lower layer based on the modified information.

3. (Once Amended) A metadata model transformer as claimed in claim- 1 further comprising:

a lower layer transformation having:

means for obtaining information from ~~of~~ a ~~lower abstraction~~ model objects in ~~from~~ the lower layer;

means for determining a specific feature included in the obtained information; and

means for creating a new ~~lower abstraction~~ model object in the lower

layer based on the specific feature.

4. (Once Amended) A metadata model transformer as claimed in claim 1 further comprising:

a lower layer transformation having:

means for obtaining relationship information between multiple ~~lower abstraction~~ model objects in ~~from~~ the lower layer; and

means for creating a new ~~lower abstraction~~ model object in the lower layer based on the relationship information.

5. (Once Amended) A metadata model transformer as claimed in claim 1 further comprising:

a higher layer transformation having:

means for obtaining information from ~~of~~ a ~~higher abstraction~~ model object in ~~from~~ the higher layer;

means for modifying the obtained information; and

means for transforming the ~~higher abstraction~~ model object in the higher layer based on the modified information.

6. (Once Amended) A metadata model transformer as claimed in claim 1 further comprising:

a higher layer transformation having:

means for obtaining information of a ~~higher abstraction~~ model objects ~~from~~ in the higher layer;

means for determining a specific feature included in the obtained information; and

means for creating a new ~~higher abstraction~~ model object in the higher layer based on the specific feature.

7. (Once Amended) A metadata model transformer as claimed in claim 1 further comprising:

a higher layer transformation having:

means for obtaining relationship information between multiple ~~higher~~

~~abstraction~~ model objects in ~~from~~ the higher layer; and

means for creating a new ~~higher abstraction~~ model object in the higher layer based on the relationship information.

8. (Once Amended) A metadata model transformer as claimed in claim 1 further comprising:

a higher layer transformation having:

means for selecting a subset of the ~~higher abstraction~~ model objects in ~~from~~ the higher layer; and

means for creating a new ~~higher abstraction~~ model object in the higher layer based on the selected subset of the ~~higher abstraction~~ model objects in the higher layer.

9. (Once Amended) A metadata model transformer for transforming a metadata model that ~~represent~~ represents one or more data sources having physical data, the metadata model having a data access layer containing data access model objects, a business layer containing business model objects, and a package layer containing package model objects, the transformation comprising:

one or more data access model transformations for refining description of the physical data in the data source expressed by the data access model objects;

one or more data access to business model transformations for constructing business model objects based on the data access model objects;

one or more business model transformations for refining the business rules expressed by the business model objects; and

one or more business to package model transformations for constructing package model objects based on the business model objects.

10. A metadata model transformer as claimed in claim 9, wherein the data access model transformations refines the description by adding new data access model objects to data access model objects which are constructed via import from the data sources or one or more metadata sources.

11. (Once Amended) A metadata model transformer as claimed in claim 9,

wherein the business model transformations ~~refines~~ refine the business rules by changing the business model objects.

12. A metadata model transformer as claimed in claim 11, wherein the business model objects include business model objects which are constructed via import from one or more metadata sources.

13. (Once Amended) A metadata model transformer as claimed in claim 9 further comprising:

one or more package model transformations for constructing a new package ~~layer~~ model object based on the package model objects in the model.

14. A metadata model transformer as claimed in claim 13, wherein the package model objects include package model objects which are constructed via import from one or more metadata sources.

15. A metadata model transformer as claimed in claim 9 further comprising:

a name mutation transformation for changing names of objects in the model based on user defined rules.

16. A metadata model transformer as claimed in claim 9, wherein the data access model transformations include a transformation which creates a new data access model object based on the data access model objects contained in the data access layer.

17. A metadata model transformer as claimed in claim 16, wherein

the data sources contain tables having columns and indexes;

the data access model objects include data access tables, data access columns and data access indexes which respectively describe information about the tables, columns and indexes in the data sources; and

the data access model transformations include a data access join constructing transformation for constructing a data access join between data access tables based on the data access indexes.

18. A metadata model transformer as claimed in claim 16, wherein

the data sources contain tables having columns and indexes;

the data access model objects include data access tables, data access columns and data access indexes which respectively describe information about the tables, columns and indexes in the data sources; and

the data access model transformations include a data access key constructing transformation for creating a data access key for a data access table based on the data access indexes.

19. A metadata model transformer as claimed in claim 16, wherein

the data sources contain at least one of tables having columns and indexes, views having columns or files having columns or fields;

the data access model objects include at least one of data access tables, data access views, data access files, data access columns and data access indexes which respectively describe information about the tables, columns of the tables, indexes of the tables, the views, the columns of the views, the files, and the columns or fields of the files in the data sources; and

the data access model transformations include a table extract constructing transformation for constructing a table extract based on the data access tables, the data access views and the data access files.

20. A metadata model transformer as claimed in claim 16, wherein

the data access model objects include one or more logical cube, each of which defines a multidimensional space represented in a number of physical storage formats; and

the data access model transformations include a data access cube constructing transformation for constructing data access cubes to instantiate the multidimensional space defined by each logical cube.

21. A metadata model transformer as claimed in claim 9, wherein the data access to business model transformations include a basic business model constructing transformation which obtains information about a data access model object in the

data access layer, and create a business model object corresponding to the data access model object.

22. (Once Amended) A metadata model transformer as claimed in claim 21, wherein

the business model objects include entities that exist as an implementation artifact of a many to many relationship, and ~~many to many~~ business joins associated with the entities; and

the business model transformations include a many to many join relationship fixing transformation for locating the entities, and replacing the associated ~~many to many~~ business joins with a single business join.

23. A metadata model transformer as claimed in claim 21, wherein

the business model objects include entities that are related via a 1:1 join relationship; and

the business model transformations include an entity coalescing transformation for locating the entities that are related via a 1:1 join relationship, and coalescing the located entities into a single entity.

24. A metadata model transformer as claimed in claim 21,

the business model objects include one or more redundant joins that express the transitivity of two or more other join relationships in the business layer; and

the business model transformations include a redundant join relationship eliminating transformation for locating the redundant joins, and eliminating the redundant joins from the business layer.

25. A metadata model transformer as claimed in claim 21, wherein

the business model transformations include a subclass relationship introducing transformation for introducing a new entity with a subclass relationship into the business layer.

26. A metadata model transformer as claimed in claim 21, wherein

the business model objects include an entity acting as a lookup table with

respect to the other entity, and a business join between the entities, the business join is an associate type; and

the business model transformations include an entity referencing transformation for locating the entity acting as a lookup table, and changing the business join which is an association type to a business join which is a reference type.

27. A metadata model transformer as claimed in claim 21, wherein

the business model transformations include an attribute usage determining transformation for determines the usage of an attribute based on how it is used by other business model objects.

28. A metadata model transformer as claimed in claim 21, wherein

the business model transformations include a date usage identifying transformation for examining attributes to determine where dates are used in the attributes.

29. A metadata model transformer as claimed in claim 9, wherein the business to package model transformations include a basic package model constructing transformation for constructing a package layer by forming a package with package model objects which corresponds to a subset of the business model objects.

30. A metadata model transformer as claimed in claim 13, wherein the package model transformations include a special package construction transformation for constructing a specific package which is usable by a specific client application from a generic package.

31. A metadata model transformer as claimed in claim 9 further comprising one or more multidimensional model transformations for a multidimensional model.

32. A metadata model transformer as claimed in claim 31, wherein the multidimensional model transformations include a measure identifying and measure dimension constructing transformation for analyzing the structure of each data

source to identify entities that contain measure candidates and identifying a reasonable set of measures.

33. A metadata model transformer as claimed in claim 31, wherein the multidimensional model transformations include a category dimension and level constructing transformation for analyzing each data source, and constructing dimensions and levels for the source model.

34. A metadata model transformer as claimed in claim 32, wherein the multidimensional model transformations include a logical cube constructing transformation for constructing a set of logical cubes based on the dimensions in a corresponding data source.

35. (Once Amended) A method for transforming a metadata model for containing model objects, the metadata model having multiple layers including a lower layer containing one or more ~~lower abstraction~~ model objects having a lower degree of abstraction ~~level~~ and a higher layer containing one or more ~~higher abstraction~~ model objects having a higher degree of abstraction ~~level~~, the method comprising steps of:

obtaining information from ~~of~~ a ~~lower abstraction~~ model object in ~~from~~ the lower layer;

abstracting the information by adding business intelligence; and

creating a model object in the higher layer ~~a higher abstraction model object~~ corresponding to the ~~lower abstraction~~ model object in the lower layer.

36. A method for transforming a metadata model that represent one or more data sources having physical data, the metadata model having a data access layer containing data access model objects, a business layer containing business model objects, and a package layer containing package model objects, the method comprising steps of:

refining description of physical data in the data sources expressed by the data access objects;

constructing business model objects based on the data access objects;

refining business rules expressed by the business model objects; and

constructing package model objects based on the business model objects.

37. A method as claimed in claim 36, wherein the step of refining the description comprises a step of adding new data access model objects to data access model objects which are constructed via import from the data sources or one or more metadata sources.

38. A method as claimed in claim 36, wherein the step of refining the business rules comprises a step of changing the business model objects.

39. A method as claimed in claim 36, wherein the step of refining the business rules uses the business model objects that include business model objects which are constructed via import from one or more metadata sources.

40. (Once Amended) A method as claimed in claim 36, further comprising a step of constructing a new package layer based on the ~~package~~ business model objects in the model.

41. (Once Amended) A method as claimed in claim 40, wherein the step of constructing a new package layer uses the ~~package~~ business model objects that include ~~package~~ business model objects which are constructed via import from one or more metadata sources.

42. A method as claimed in claim 36 further comprising a step of changing names of objects in the model based on user defined rules.

43. (Once Amended) A computer readable memory for storing code which identifying instructions for transforming a metadata model for containing model objects, the metadata model having multiple layers including a lower layer containing one or more ~~lower abstraction~~ model objects having a lower degree of abstraction ~~level~~ and a higher layer containing one or more ~~higher abstraction~~ model objects having a higher degree of abstraction ~~level~~, the instructions comprising:

obtaining information from ~~of~~ a ~~lower abstraction~~ model object in ~~from~~ the

lower layer;

abstracting the information by adding business intelligence; and

creating a model object in the higher layer ~~a higher abstraction model object~~ corresponding to the ~~lower abstraction~~ model object in the lower layer.

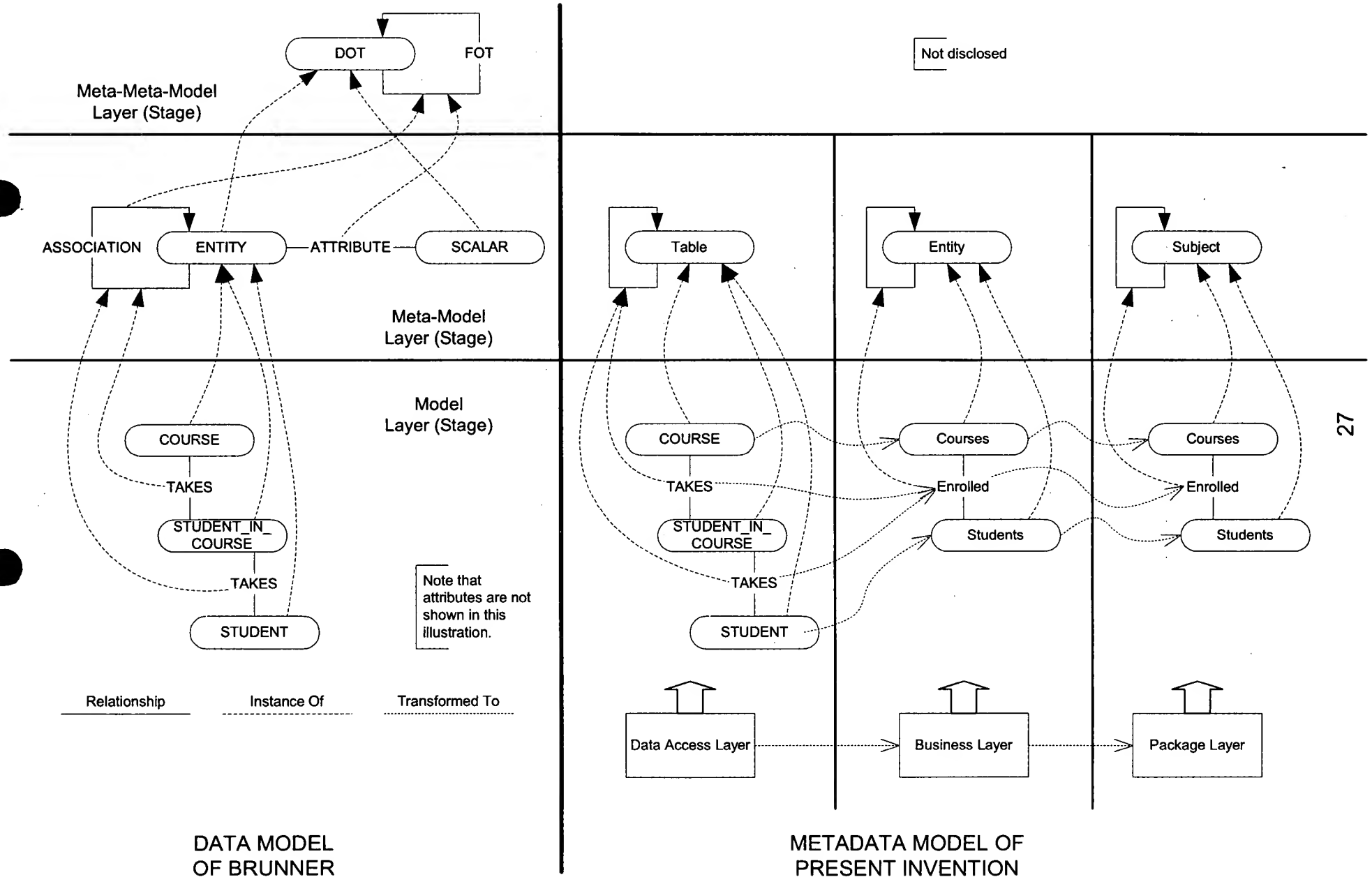
44. (Once Amended) A computer data signal representing code which identifying instructions for transforming a metadata model for containing model objects, the metadata model having multiple layers including a lower layer containing one or more ~~lower abstraction~~ model objects having a lower degree of abstraction ~~level~~ and a higher layer containing one or more ~~higher abstraction~~ model objects having a higher degree of abstraction ~~level~~, the instructions comprising:

obtaining information from ~~of~~ a ~~lower abstraction~~ model object in ~~from~~ the lower layer;

abstracting the information by adding business intelligence; and

creating a model object in the higher layer ~~a higher abstraction model object~~ corresponding to the ~~lower abstraction~~ model object in the lower layer.

APPENDIX A


DOT
FOT
Meta-Meta-Model Layer (Stage)
Not disclosed
ASSOCIATION
ENTITY
ATTRIBUTE
SCALAR
Table
Entity
Subject
Meta-Model Layer (Stage)
Model Layer (Stage)
COURSE
TAKES
STUDENT_IN_ COURSE
TAKES
STUDENT
Note that attributes are not shown in this illustration.
COURSE
TAKES
STUDENT_IN_ COURSE
TAKES
STUDENT
Courses
Enrolled
Students
Courses
Enrolled
Students
Relationship
Instance Of
Transformed To
Data Access Layer
Business Layer
Package Layer
DATA MODEL OF BRUNNER
METADATA MODEL OF PRESENT INVENTION